AMENDMENTS TO THE CLAIMS

1. (Currently Amended) In a computing device, a method comprising:

providing a first function in a first programming language;

~~providing~~ deriving a definition of ~~a~~the first function ~~associated with a first language~~;

creating description information about the first function ~~from~~ using the definition of ~~a~~the first function ~~associated with a first language~~;

~~storing the description information in a storage device;~~

~~providing a call to the function in the first language;~~

~~retrieving the description information for the function from the storage device; and~~

translating ~~the call to~~ the first function in the first programming language into ~~a call to a~~ corresponding function in a second programming language using the definition of the first function~~description information, wherein translating avoids processing the definition of the function for each translation~~;

generating a function library containing the corresponding function in the second programming language and the description information;

storing the function library;

translating a first file in the first programming language to a corresponding file in the second programming language, the first file comprising one or more calls to the first function, the translating comprising:

accessing the description information about the first function for each of the one or more calls to the first function; and

using the description information to create each call to the corresponding function in the second programming language while avoiding accessing the first function.

2. (Original) The method of claim 1, further comprising: storing the description information in a file of description items.

3. (Currently Amended) The method of claim 1, wherein creating description information about the function comprises: examining the definition of the function associated with the first programming language; and deriving information about the function.

4. (Cancelled)

5. (Currently Amended) The method of claim 1, further comprising: storing a translated function in the second programming language in a library of entries.

6. (Previously Presented) The method of claim 1, in which creating description information about the function comprises: deriving a number of declared formal inputs to the function.

7. (Previously Presented) The method of claim 1, in which creating description information about the function comprises: deriving a number of declared formal outputs to the function.

8. (Previously Presented) The method of claim 1, in which creating description information about the function comprises: deriving a scope of the function.

9. (Previously Presented) The method of claim 1, in which creating description information about the function comprises: determining whether the function accepts a variable number of arguments.

10. (Previously Presented) The method of claim 1, in creating description information about the function comprises: determining whether the function returns a variable number of results.

11. (Currently Amended) In a computing device, a method comprising:

providing a file of description items, each item including description information about a function associated with a first programming language,

providing a first program file in the first programming language, the first program file containing one or more~~a~~ calls to the function in the first programming language;

retrieving an item from the file of description items;

using the description information to translate each of the one or more ~~a~~ calls to the function in the first programming language into a call to a corresponding function in a second programming language in a manner that avoids accessing the function in the first programming

language ~~, wherein translation avoids processing of the definition of the function for each translation~~; and

using the file of description items to translate ~~a~~ the first program file into ~~a~~ the second program file.

12. (Original) The method of claim 11, wherein the description information about the function comprises: a descriptor identifying a declared number of formal inputs to the function.

13. (Original) The method of claim 11, wherein the description information about the function comprises: a descriptor identifying a declared number of formal outputs to the function.

14. (Original) The method of claim 11, wherein the description information about the function comprises: a descriptor identifying a scope of the function.

15. (Original) The method of claim 11, wherein the description information about the function comprises: a descriptor identifying an acceptance of a variable input argument list into the function.

16. (Original) The method of claim 11, wherein the description information about the function comprises: a descriptor identifying a return of a variable output argument list from the function.

17. (Previously Presented) The method of claim 11, further comprising:

storing the translated call in the second program file.

18. (Original) The method of claim 11, wherein using the file of description items comprises: generating a call through a function evaluation interface for the function if the description information includes a descriptor identifying an acceptance of a variable input argument list into the function.

19. (Original) The method of claim 11, wherein using the file of description items comprises: generating a call through a function evaluation interface for the function if the description information includes a descriptor identifying a return of a variable output argument list from the

function.

20. (Original) The method of claim 11, wherein using the file of description items comprises: generating a call through a normal interface for the function if the description information includes a descriptor identifying a known number of input and output arguments to the function.

21. (Currently Amended) In a computing device, a method comprising:

providing a library file including functions defined by a first programming language;

creating a function library and a description file from the library file, the function library including one or more functions defined by a second language, each function in the function library being a translated version of a function in the library file, and the description file including description information about each function in the library file, ~~wherein~~ the description information ~~enables~~ enabling translation of a call to the function in the first programming language into a call to a corresponding function in the second programming language ~~, wherein translation avoids~~ while avoiding ~~processing of the definition of~~accessing the function in the first programming language for each translation;

~~providing a call to the function in the first language;~~

~~retrieving the description file;~~ and

using the description file to translate a program file from the first programming language into the second programming language, ~~wherein~~ the program file in the first programming language comprising one or more calls to a function in the library file, translating each call ~~in the program file~~to the function in the first programming language ~~to a function in the library file is translated~~ into a call to a corresponding function in the second programming language.

22. (Currently Amended) The method of claim 21, wherein creating a function library comprises: translating the call to each function in the library file into a call to a corresponding function in the second programming language.

23. (Previously Presented) The method of claim 21, wherein creating a description file comprises:

examining the definition of each function in the library file; and

deriving information about each function.

24. (Previously Presented) The method of claim 23, further comprising: using the derived information about each function to create description information; and creating a description file including description information about each function in the library file.

25. (Currently Amended) The method of claim 21, wherein using the description file comprises: for each call in the program file to a function in the library file, retrieving the description information about the function from the description file; and using the description information to translate the call to the function in the first programming language into a call to a corresponding function in the second programming language.

26. (Original) The method of claim 21, wherein using the description file comprises: generating a call through a function evaluation interface for the function if the description information includes a descriptor identifying an acceptance of a variable input argument list into the function.

27. (Original) The method of claim 21, wherein using the description file comprises: generating a call through a function evaluation interface for the function if the description information includes a descriptor identifying a return of a variable output argument list from the function.

28. (Original) The method of claim 21, wherein using the description file comprises: generating a call through a normal interface for the function if the description information includes a descriptor identifying a known number of input and output arguments to the function.

29. (Currently Amended) A computer program product, tangibly stored on a computer-readable medium, for creating a data file, the product comprising instructions operable to cause a programmable processor to:

obtain a first function in a first programming language;

derive ~~obtain~~ a definition of the first ~~a~~ function ~~associated with a first language~~;

create description information about the first function using ~~from~~ the definition of the first function ~~associated with a first language~~;

~~store the description information in a storage device;~~

~~provide a call to the function in the first language;~~

~~retrieve the description information for the function from the storage device; and~~

translate the first function in the first programming language into a corresponding function in a second programming language using the definition of the first function;

generate a function library containing the corresponding function in the second programming language and the description information;

store the function library;

translate a first file in the first programming language to a corresponding file in the second programming language, the first file comprising ~~the~~one or more calls to the first function, said translating causing the processor to:

access the description information about the first function for each of the one or more calls to the first function; and

use the description information to create ~~in the first language into~~ a call to ~~a~~ the corresponding function in ~~a~~ the second programming language ~~using the description information, wherein translation avoids~~ while avoiding ~~processing the definition of~~accessing the first function ~~for each translation~~.

30. (Original) The product of claim 29, further comprising instructions operable to cause a programmable processor to: store the description information in a file of description items.

31. (Currently Amended) The product of claim 29, wherein creating description information comprises: examining the definition of the function associated with the first programming language; and deriving information about the function.

32. (Original) The product of claim 31, further comprising instructions operable to cause a programmable processor to: use the derived information to create the description information.

33. (Cancelled)

34. (Previously Presented) The product of claim 29, in which creating description information comprises: deriving a number of declared formal inputs to the function.

35. (Previously Presented) The product of claim 29, in which creating description information comprises: deriving a number of declared formal outputs to the function.

36. (Previously Presented) The product of claim 29, in which creating description information comprises: deriving a scope of the function.

37. (Previously Presented) The product of claim 29, in which creating description information comprises: determining whether the function accepts a variable number of arguments.

38. (Previously Presented) The product of claim 29, in which creating description information comprises: determining whether the function returns a variable number of results.

39. (Currently Amended) A product, stored on a machine-readable medium, for translating a program file, the product comprising instructions operable to cause a processor to:

provide a file of description items, each item including description information about a function associated with a first programming language,

provide a first program file in the first programming language, the first program file comprising one or more ~~a~~ calls to the function in the first programming language;

retrieve an item from the file of description items;

use the description information to translate each of the one or more calls to the function in the first programming language into a call to a corresponding function in a second programming language~~, wherein translation avoids~~ while avoiding ~~processing of the definition of~~accessing the function in the first programming language ~~for each translation~~; and

use the file of description items to translate a first program file into a second program file.

40. (Original) The product of claim 39, wherein the description information about the function comprises: a descriptor identifying a declared number of formal inputs to the function.

41. (Original) The product of claim 39, wherein the description information about the function comprises: a descriptor identifying a declared number of formal outputs to the function.

42. (Original) The product of claim 39, wherein the description information about the function comprises: a descriptor identifying a scope of the function.

43. (Original) The product of claim 39, wherein the description information about the function comprises: a descriptor identifying an acceptance of a variable input argument list into the function.

44. (Original) The product of claim 39, wherein the description information about the function comprises: a descriptor identifying a return of a variable output argument list from the function.

45. (Previously Presented) The product of claim 39, further comprising instructions operable to cause a programmable processor to:

store the translated call in the second program file.

46. (Original) The product of claim 39, wherein using the file of description items comprises: generating a call through a function evaluation interface for the function if the description information includes a descriptor identifying an acceptance of a variable input argument list into the function.

47. (Original) The product of claim 39, wherein using the file of description items comprises: generating a call through a function evaluation interface for the function if the description information includes a descriptor identifying a return of a variable output argument list from the function.

48. (Original) The product of claim 39, wherein using the file of description items comprises: generating a call through a normal interface for the function if the description information includes a descriptor identifying a known number of input and output arguments to the function.

49. (Currently Amended) A computer program product, tangibly stored on a computer-readable medium, for translating function calls, the product comprising instructions operable to cause a programmable processor to:

provide a library file including functions defined by a first programming language;

create a function library and a description file from the library file, the function library including one or more functions defined by a second programming language, each function in the function library being a translated version of a function in the library file, and the description file including description information about each function in the library file, ~~wherein~~ the description information ~~enables~~ enabling translation of a call to the function in the first programming language into a call to a corresponding function in the second programming language ~~, wherein the translation avoids processing of the definition of~~ in a manner that avoids accessing the function in the first programming language for each translation;

~~provide a call to the function in the first language;~~

~~retrieve the description file;~~ and

use the description file to translate a program file from the first programming language into the second programming language, ~~wherein~~ the program file in the first programming language comprising one or more ~~each~~ calls ~~in the program file~~ to a function in the library file, translating ~~is translated~~ each call to the function in the first programming language into a call to a corresponding function in the second programming language.

50. (Currently Amended) The product of claim 49, wherein creating a function library comprises: translating the call to each function in the library file into a call to a corresponding function in the second programming language.

51. (Original) The product of claim 49, wherein creating a description file comprises: examining the definition of each function in the library file; and deriving information about each function.

52. (Original) The product of claim 51, further comprising: using the derived information about each function to create the description information; and creating a description file including description information about each function in the library file.

53. (Original) The product of claim 49, wherein using the description file comprises: for each call in the program file to a function in the library file, retrieving the description information about the function from the description file; and using the description information to translate the call to the function in the first language into a call to a corresponding function in the second

language.

54. (Original) The product of claim 49, wherein using the description file comprises: generating a call through a function evaluation interface for the function if the description information includes a descriptor identifying an acceptance of a variable input argument list into the function.

55. (Original) The product of claim 49, wherein using the description file comprises: generating a call through a function evaluation interface for the function if the description information includes a descriptor identifying a return of a variable output argument list from the function.

56. (Original) The product of claim 49, wherein using the description file comprises: generating a call through a normal interface for the function if the description information includes a descriptor identifying a known number of input and output arguments to the function.